



UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.		
10/659,762	09/10/2003	Bruce W. Warila	2008311-0001	4844		
24280	7590	04/08/2008				
CHOATE, HALL & STEWART LLP		EXAMINER				
TWO INTERNATIONAL PLACE		TECKLU, ISAAC TUKU				
BOSTON, MA 02110		ART UNIT	PAPER NUMBER			
		2192				
NOTIFICATION DATE	DELIVERY MODE					
04/08/2008	ELECTRONIC					

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Notice of the Office communication was sent electronically on above-indicated "Notification Date" to the following e-mail address(es):

patentdocket@choate.com

Office Action Summary	Application No.	Applicant(s)
	10/659,762	WARILA ET AL.
	Examiner ISAAC T. TECKLU	Art Unit 2192

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --
Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If no period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

- 1) Responsive to communication(s) filed on 21 December 2007.
- 2a) This action is FINAL. 2b) This action is non-final.
- 3) Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

- 4) Claim(s) 1-48, 50, 52, 70-73, 75-87, 89 and 91-108 is/are pending in the application.
 - 4a) Of the above claim(s) _____ is/are withdrawn from consideration.
- 5) Claim(s) _____ is/are allowed.
- 6) Claim(s) 1-48, 50, 52, 70-73, 75-87, 89 and 91-108 is/are rejected.
- 7) Claim(s) _____ is/are objected to.
- 8) Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

- 9) The specification is objected to by the Examiner.
- 10) The drawing(s) filed on _____ is/are: a) accepted or b) objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).
- 11) The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

- 12) Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).
 - a) All b) Some * c) None of:
 1. Certified copies of the priority documents have been received.
 2. Certified copies of the priority documents have been received in Application No. _____.
 3. Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

- 1) Notice of References Cited (PTO-892)
- 2) Notice of Draftsperson's Patent Drawing Review (PTO-948)
- 3) Information Disclosure Statement(s) (PTO/SB/08)
Paper No(s)/Mail Date _____
- 4) Interview Summary (PTO-413)
Paper No(s)/Mail Date. _____
- 5) Notice of Informal Patent Application
- 6) Other: _____

DETAILED ACTION

1. This action is responsive to the amendment filed on 12/21/2007.
2. Claims 49, 51, 53-69, 74, 88 and 90 have been cancelled.
3. Claims 1-48, 50, 52, 70-73, 75-87, 89 and 91-108 have been reexamined.

Oath/Declaration

4. The oath or declaration is defective. A new oath or declaration in compliance with 37 CFR 1.67(a) identifying this application by application number and filing date is required. See MPEP §§ 602.01 and 602.02.

The oath or declaration is defective because:

It does not identify the mailing address of each inventor. A mailing address is an address at which an inventor customarily receives his or her mail and may be either a home or business address. The mailing address should include the ZIP Code designation. The mailing address may be provided in an application data sheet or a supplemental oath or declaration. See 37 CFR 1.63(c) and 37 CFR 1.76.

It was not executed in accordance with either 37 CFR 1.66 or 1.68.

Claim Objections

5. Claims 14-15, 21, 26, 28, etc. recite acronym "OS", such acronym should be spelled out once in the claims as its intended meaning and utility is likely to be changed over the times.

Appropriate correction is required.

6. Claims 95, 99, 104 and 108 recite acronym "XML", such acronym should be spelled out once in the claims as its intended meaning and utility is likely to be changed over the times.

Appropriate correction is required.

7. Claim 7, recites "the method of claim 1 further further comprising:" (line1). The word further is redundant. Appropriate correction is required.

8. Claim 35, recites "the method of any of claim 1" (in line 1). It should be written as 'the method of claim 1". Similarly claims 45-46 are objected for the same issue above. Appropriate correction is required.

9. Claim 50, recites "the method of any of claims 47" (in line 1). It should be written as 'the method of claim 47". Similarly claim 52 is objected for the same issue above. Appropriate correction is required. Applicant is required to proofread the whole application before filling a response.

Claim Rejections - 35 USC § 112

10. The following is a quotation of the second paragraph of 35 U.S.C. 112:

The specification shall conclude with one or more claims particularly pointing out and distinctly claiming the subject matter which the applicant regards as his invention.

11. Claims 20 and 24 are rejected under 35 U.S.C. 112, second paragraph, as being indefinite for failing to particularly point out and distinctly claim the subject matter which applicant regards as the invention.

Claim 20 recites the limitation "the remote device" in line 12. There is insufficient antecedent basis for this limitation in the claim.

Claim 24 recites the limitation "the running state and functionality of an application" in line 2. There is insufficient antecedent basis for this limitation in the claim.

Claim Rejections - 35 USC § 103

12. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

(a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary

skill in the art to which said subject matter pertains. Patentability shall not be negated by the manner in which the invention was made.

13. Claims 1-48, 50, 52, 70-73, 75-76, 79-87, 89 and 91-108 are rejected under 35 U.S.C. 103(a) as being unpatentable over Lewallen (US 6,854,123 B1) in view of Geric et al. (US 2003/0046316 A1).

Per claim 1(Currently amended), xyz discloses a method for enabling the creation and management of a platform-independent applications whose appearance and functionality is consistently propagated across heterogeneous device types for cross-device interoperability, replicability, and compatibility of applications and data with a consistent user experience (col. 1:45-60 "... platform independent appearance and standard behavior..."), the method comprising:

— generating receiving, by a device, a platform independent data superstructure defining the appearance and behavior of an application independent of characteristics of a digital processing device on which the application is to be instantiated, (col. 8:60-67 "... data in any object accessible to the user interface including DOM tree..." and col.9:55-70 "... executing mixed statement ..." and e.g. FIG. 3a, step 100 and related text) the platform-independent data superstructure storing an application state, program code and internal logic of the application (e.g. FIG. 3a, step 108, 130 and 134 "method in native object class to determine pointer to node info and pointer to UI object in node info" and related text)

instantiating by a superstructure-dedicated operating system the application in the device in accordance with the received superstructure (e.g. FIG. 3a, step 108 “API function to instantiate a new java object” and related text);

receiving, by the platform-independent data superstructure (col.3:20-30 “... platform independent ... to allow Java application access operating system ...”), from a device-native operating system via communication with the superstructure-dedicated operating system at least one application event generated by the instantiated application and representative of an update to the application state of the application (col. 5:45-50 “... UI objects to native operating system ...” and col.5:55-65 “... transform the user interface APIs and objects to native operating system objects ...”)

updating, by the platform-independent data superstructure, in-response to generated application events information in a first segment of the superstructure associated with the at least one application events (col.6:15-25 “...mappings to transform W3C API interface in the mixed statement program ...”), responsive to receiving the at least one application event and independent of an update to a second segment in the superstructure the application events including events generated by the application instantiated in the device and representative of an application state, and

updating, in accordance with the superstructure segment update, the application state in the device (col. 12:40-50 “... convert source code statement ... capable of executing multiple operating system ...” and e.g. FIG. 3b, step 118 and related text). Wherein;

Lewallen does not explicitly disclose updating the application state in the device. However, Gergic discloses a new application programming language which is based on user

interaction with any device which a user is employing to access any type of information. Geric teaches When the transcending is performed by a multi-modal/conversational browser (as described below), the gestures are uniquely identified using a `node_id` tag. This allows not only to produce the rendering in each registered modality (local or distributed), but also to provide very tight synchronization (i.e., on a gesture level or even sub-gestures levels, when it is a gesture for which this makes sense). For example, an event (I/O event) immediately impacts the state of the dialogs (i.e., the state as maintained in the multi-modal shell,. In addition to the above Geric teaches that upon successfully executing an update, the shell passes the newly updated application state to all registered browsers (paragraph [1312] and update the application state as each item of information is furnished (paragraph [1317]). Therefore it would have been obvious to one skilled in the art at the time the invention was made to update the application state to extract relevant information that can be present to the user during user interaction (paragraph [1321]).

Per claim 2 (Currently amended), xyz discloses the method of claim 1, wherein the step of receiving further comprises receiving, by the device, a platform-independent, hierarchical information superstructure defining the appearance and behavior of an application (col. 1:45-60 “... platform independent appearance and standard behavior...”) and storing an application state, program code and internal logic of the application (e.g. FIG. 3a, step 108, 130 and 134 “method in native object class to determine pointer to node info and pointer to UI object in node info” and related text).

Per claim 3 (Currently amended), Lewallen discloses the method of claims 1 or 2 further comprising:

generating receiving, by the device, a message containing a data object of a defined type operable to instantiate the application in the device and, (e.g. FIG. 3a, step 108 “API function to instantiate a new java object” and related text),

instantiating the application in the device in accordance with the data object in the received message (col. 7:25-40 “... instantiate a new object ...”).

Per claim 4 (Currently amended), Lewallen discloses the method of claim 1 wherein the instantiating of the superstructure application inside the target device occurs substantially when the application is invoked (e.g. FIG. 3a, step 108 “API function to instantiate a new java object” and related text).

Per claim 5 (Currently amended), Lewallen discloses the method of claim 1 wherein the instantiating of the application inside the target device occurs at an application provisioning time prior to application run-time (e.g. FIG. 3a, step 108 “API function to instantiate a new java object” and related text).

Per claim 6 (Currently amended), Lewallen discloses the method of claim 1 further wherein:

identifying, by a provisioning application on a first device locates within its operating environment a first superstructure for a new application superstructure to be expressed to a second device (e.g. FIG. 1, 18 and related text);
generating by, the provisioning application generates a defined data object to be used to for express the new application superstructure to the second device (e.g. FIG. 3b, 122 and related text);
transmitting, to the second device, the data object (e.g. FIG. 1, 6 and related text); and
creating by the second device the new application superstructure responsive to receiving the data object (e.g. FIG. 3a, step 136 and related text).

Per claim 7 (Currently amended), Lewallen discloses the method of claim 1 further comprising the steps of:

identifying by a provisioning application on a first device within an operating environment on the first device, a predefined data object that expresses a new application superstructure for a second device (e.g. FIG. 1, 18 and related text);
transmitting, to the second device, the predefined data object (e.g. FIG. 1, 6 and related text); and
creating by the second device superstructure from the data object in the message (e.g. FIG. 3a, step 136 and related text).

Per claim 8 (Currently amended), Lewallen discloses the method of claim 1 further comprising:

maintaining by a first device an application capable of accepting input from a user to create an interactive message (e.g. FIG. 3a, step 136 and related text);

translating by the first application an operational portion of the message into a new superstructure-based application operable to display the message or cause interactive operations within the message (col. 11:60-70 "... translate the application program ..."); and

transmitting by the first application the superstructure of the new application to a receiving device (e.g. FIG. 1, 6 and related text).

Per claim 9 (Currently amended), Lewallen discloses the method of claim 8 further comprising: the step of converting the superstructure into a temporary form that is transmitted, received (col. 12:40-50 "... convert source code statement ... capable of executing multiple operating system ..." and e.g. FIG. 3b, step 118 and related text), and decoded back into an original form on the receiving device; and

maintaining, by the second device, an application that receives the superstructure in its temporary form, decodes it, and causes the message-bearing superstructure to operate, thereby rendering the message (e.g. FIG. 3a, step 136 and related text).

Per claim 10 (Currently amended), Lewallen discloses the method of claims 1 wherein the step of receiving further comprises receiving, by the device, for a given state of a selected application, a platform-independent data superstructure having a substantially invariant organization regardless of the device, platform or device-native operating system environment in which the associated application is instantiated (col. 1:45-60 "... platform independent appearance and standard behavior..."), so as to maintain a consistent application

appearance and behavior across heterogeneous devices, platforms or device-native operating system environments (col. 8:60-67 "... data in any object accessible to the user interface including DOM tree..." and e.g. FIG. 3a, step 100 and related text).

Per claim 11 (Currently amended), Lewallen discloses the method of claims 1 wherein the step of receiving further comprises receiving by the device, a platform-independent data superstructure defining plurality of rules of appearance and behavior of the application which are substantially invariant across heterogeneous devices, platforms or device-native operating system environments (col. 1:45-60 "... platform independent appearance and standard behavior...").

Per claim 12 (Currently amended), Lewallen discloses the method of claims 1 wherein the step of receiving further comprises receiving by the device, substantially identical application source code in the platform-independent data superstructure as a source code across heterogeneous devices, platforms or device-native operating system environment (col. 1:45-60 "... platform independent appearance and standard behavior...").

Per claim 13 (Currently amended), Lewallen discloses the method of claims 1 wherein the step of initiating the application further comprises initiating by the superstructure dedicated operating system an application including a user interface having a substantially identical appearance and behavior across heterogeneous devices, platforms or device-native

operating system environments (e.g. FIG. 1, 18 and col. 1:45-60 "... platform independent appearance and standard behavior...").

Per claim 14 (Currently amended), Lewallen discloses the method of claim 1 wherein thee step of updating information in the segment of the superstructure further comprises the steps of:

receiving data representative of the at least one application event in the superstructure dedicated OS (e.g. FIG. 1, 6 and related text),

applying to the superstructure, in response to the received data, a data object, thereby modifying the superstructure (col.3:20-30 "... platform independent ... to allow Java application access operating system ..."),

Per claim 15 (Currently amended), Lewallen discloses the method claim 1 further comprising the steps of: generating, the superstructure-dedicated operating system a modification data object representative of the a modification to be applied to the superstructure,

translating the modification data object into a form suitable for processing by the device-native OS (col. 11:60-70 "... translate the application program ..."),

receiving in the device-native OS the translated modification data object, and processing the translated modification data object in the application to update the application (e.g. FIG. 1, Native O/S objects and interfaces and related text).

Per claim 16 (Currently amended), Lewallen discloses the method of claim 15 further comprising expressing within the superstructure a mechanism for generating the modification data object (col. 12:40-50 "... convert source code statement ... capable of executing multiple operating system ..." and e.g. FIG. 3b, step 118 and related text).

Per claim 17 (Currently amended), Lewallen discloses the method claim 14 wherein modifying the superstructure includes transmitting a portion of the superstructure to a processor remote from the device, modifying the transmitted portion, and then returning the modified portion or a new set of operations to update the superstructure (col. 11:60-70 "... translate the application program ...").

Per claim 18 (Currently amended), Lewallen discloses the method of claim 14 wherein modifying the superstructure includes using device-native code to implement an interface to modify the superstructure (col. 12:40-50 "... convert source code statement ... capable of executing multiple operating system ..." and e.g. FIG. 3b, step 118 and related text).

Per claim 19 (Currently amended), Lewallen discloses the method of claim 14 wherein the application of changes to the superstructure is implemented by activating program instructions within the superstructure (e.g. FIG. 2, 32 260 and related text).

Per claim 20 (Currently amended), Lewallen discloses the method of claim 1 wherein further comprising the step of:

storing by an application server in communication with the device (e.g. FIG. 1, 24 and related text)

a copy of platform-independent data superstructure the superstructure including at least one data objects operable to instantiate applications on the device and, transmitting application from the application server to the device providing communication of applications between the

application server and the remote device by replicating data objects in the superstructure to the remote device via the communications channel, so as to enable instantiation of new data objects and applications from the server into the remote device (e.g. FIG. 1, 16 and related text).

transmitting application from the application server to the device replicating data objects in the superstructure to the remote device, so as to enable instantiation of new data objects and applications from the server into the remote device (e.g. FIG. 1 and related text).

Per claim 21 (Currently amended), Lewallen discloses a method for enabling the creation and management of platform-independent applications whose appearance and functionality is consistently propagated across heterogeneous device types for cross-device interoperability, replicability, and compatibility of applications and data with a consistent user experience (col. 1:45-60 "... platform independent appearance and standard behavior..."), the method comprising:

generating receiving by a device a platform-independent data superstructure defining the appearance and behavior of an application, the superstructure storing an application state, program code and internal logic of the application (col. 1:45-60 "... platform independent appearance and standard behavior..."),

instantiating by superstructure-dedicated operating system, the application in the device in accordance with the received platform-independent data superstructure (e.g. FIG. 3a, step 108 “API function to instantiate a new java object” and related text),
transmitting, to the superstructure-dedicated operating system, by a device-native operating system, at least one application event generated by the instantiated application and representative of an update to the application state of the application (e.g. FIG. 1, 6 and related text);

updating, in response to generated application events, information in a segment of the superstructure associated with the application events, the application events including events generated by the application instantiated in the device and representative of an application state, and updating, in accordance with the superstructure segment update, the application state in the device, wherein (e.g. FIG. 15, 506 “UPDATE PLATFORM INDEP. PRES. MODEL WITH RUNNING STATE” and related text);

the superstructure is a hierarchical information structure, application appearance and behavior are encapsulated within the superstructure (e.g. FIG. 11, 254 and related text), and application events are expressed to the superstructure via a pathway including a device-native operating system (OS) and a superstructure-dedicated OS acting as an intermediary between the device-native OS and the superstructure (e.g. FIG. 10, 204 and 202 and related text), whereby:

a defined portion of the application can be addressed and updated in response to application events without necessitating update of the entire application, and the appearance and behavior of the application can be propagated with consistency across heterogeneous device

~~types, to enable cross-device interoperability, replicability, and compatibility of applications and data with a consistency of user experience further wherein:~~

~~transmitting by the superstructure-dedicated operating system, to a remote server an application event is expressed to the superstructure, a segment of the superstructure associated with the at least one application event (e.g. FIG. 1, 6 and related text),~~

receiving from the server a modified version of the segment of the superstructure generated, responsive to the received segment of the superstructure for replacement of the existing version of the segment of the superstructure thus updating the segment of the superstructure (e.g. FIG. 1, 20-2c and related text) and

instructing by the superstructure-dedicated OS, the device native OS to update the application state in response to the updated segment of the superstructure (e.g. FIG. 1, 18 and 6 and related text).

Lewallen does not explicitly disclose updating the segment of the superstructure. However, Gericc discloses a new application programming language which is based on user interaction with any device which a user is employing to access any type of information. Gericc teaches When the transcending is performed by a multi-modal/conversational browser (as described below), the gestures are uniquely identified using a node_id tag. This allows not only to produce the rendering in each registered modality (local or distributed), but also to provide very tight synchronization (i.e., on a gesture level or even sub-gestures levels, when it is a gesture for which this makes sense). For example, an event (I/O event) immediately impacts the state of the dialogs (i.e., the state as maintained in the multi-modal shell,. In addition to the above

Geric teaches that upon successfully executing an update, the shell passes the newly updated application state to all registered browsers (paragraph [1312] and update the application state as each item of information is furnished (paragraph [1317]). Therefore it would have been obvious to one skilled in the art at the time the invention was made to update the application state to extract relevant information that can be present to the user during user interaction (paragraph [1321]).

Per claim 22 (Currently amended), Lewallen discloses the method of claim 21 wherein the network further comprises a plurality of heterogeneous devices, communications channels and communications providers servicing the communications channels, and wherein the superstructure defines a given application to have an appearance and behavior that can be propagated with consistency across heterogeneous devices, communications channels and communications providers (e.g. FIG. 1, 6 and related text).

Per claim 23 (Currently amended), Lewallen discloses the method of claim 21 wherein: modifying the superstructure in a substantially device-independent manner (col. 12:40-50 "... convert source code statement ... capable of executing multiple operating system ..." and e.g. FIG. 3b, step 118 and related text), and expressing a real-time image of an application running in a first device can be expressed across the network from the first device to a second device to yield a viable instantiation of the application in the second device, regardless of device environment, (col. 5:45-50 "... UI objects to native operating system ..." and col. 5:55-65 "... transform the user interface APIs and objects

to native operating system objects ...").

Per claim 24 (Currently amended), Lewallen discloses the method of claim 21 wherin:
by the superstructure the running state and functionality of an application operating in a
first device (e.g. FIG. 1, 6 and 18 and related text), and
instantiating the application on a second device, without loss of state or functionality, by
expressing the superstructure into the second device e.g. FIG. 3a, step 108, 130 and 134 "method
in native object class to determine pointer to node info and pointer to UI object in node info" and
related text).

Per claim 25, Lewallen discloses the method of claim 1 further comprising validating the
superstructure upon or after modification (e.g. FIG. 3a, step 102, 108 and related text).

Per claim 26, Lewallen discloses the method of claim 1 further comprising validating the
superstructure after modifying the superstructure, the validation including validation of data
updated by processing of an event, so that the modified superstructure cannot express a harmful
change to the device-native OS (e.g. FIG. 3a, step 102, 108 and related text).

Per claim 27 (Currently amended), Lewallen discloses the method of claim 1 further
comprising producing by an application defined by the superstructure external changes only by

invoking operations that operate on the superstructure (e.g. FIG. 1, 18 and related text).

Per claim 28, Lewallen discloses the method of claim 1 further including providing an interface between an application and a system service, wherein the interface is defined by interaction between the superstructure and the superstructure-dedicated OS (e.g. FIG. 1, 6 and 202 and related text).

Per claim 29 (Currently amended), Lewallen discloses the method of claims 1 or 21
further comprising:
expressing, by an information processing language adapted to interface with the superstructure, a
set of transformations within the superstructure, the information processing
language being expressible entirely within the superstructure and capable of expressing a set of transformations within the superstructure (col. 5:45-50 "... UI objects to native operating system ..." and col. 5:55-65 "... transform the user interface APIs and objects to native operating system objects ..."), and

modifying by the information processing language data only within the superstructure, so that: applications utilizing the language cannot affect the state of other applications or operate outside a bounded application container to affect an underlying device platform (col. 12:40-50 "... convert source code statement ... capable of executing multiple operating system ..." and e.g. FIG. 3b, step 118 and related text).

Per claim 30, Lewallen discloses the method of claim 1 wherein the superstructure can contain stylesheets for defining selected application or presentation characteristics (col. 9:35-40 “XML … document … ”).

Per claim 31, Lewallen discloses the method of claim 30 further comprising configuring stylesheets on a per-device basis (col. 9:35-40 “XML … document … ”).

Per claim 32, Lewallen discloses the method of claim 30 further comprising configuring stylesheets on a per-group-of-devices basis (col. 9:35-40 “XML … document … ”).

Per claim 33, Lewallen discloses the method of claim 30 further comprising expressing stylesheets within the superstructure, independent of device-specific limitations (e.g. FIG. 1, 24 and related text).

Per claim 34, Lewallen discloses the method of claim 30 further comprising selecting a stylesheet at runtime (col. 9:35-40 “XML … document … ”).

Per claim 35 (Currently amended), Lewallen discloses the method of any of claim 1 further comprising the step of transmitting an application defined by the superstructure via a peer to peer transaction from a first device in which the application is instantiated, to a second device for instantiation in the second device (col. 8:60-67 “… data in any object accessible to the user interface including DOM tree…” and e.g. FIG. 3a, step 100 and related text).

Per claim 36 (Currently amended), Lewallen discloses the method of claim 1, further comprising:

converting at least a portion of the superstructure into a device-portable form, independent of the present state of the application; and reconstructing the original superstructure portion, on the same or different device context, using the device portable form, without loss of state (col. 12:40-50 "... convert source code statement ... capable of executing multiple operating system ..." and e.g. FIG. 3b, step 118 and related text)

Per claim 37, Lewallen discloses the method of claim 36 wherein the reconstructing includes utilizing a new device-specific stylesheet (col. 9:35-40 "XML ... document ...").

Per claim 38 (Currently amended), Lewallen discloses a system for enabling the creation and management of platform-independent application, whose appearance and functionality is consistently propagated across heterogenous device types for cross-device interoperability, replicability, and compatibility of applications and data with a consistent user experience, comprising (col. 1:45-60 "... platform independent appearance and standard behavior...")

a platform-independent data superstructure defining the appearance and behavior of an application (col. 1:45-60 "... platform independent appearance and standard behavior...") and storing an application state, program code and internal logic of the application (e.g. FIG. 3a, step 108, 130 and 134 "method in native object class to determine pointer to node info and pointer to UI object in node info" and related text)

at least one application event generated by the application and representative of an application state: (col. 5:45-50 "... UI objects to native operating system ..." and col.5:55-65 "... transform the user interface APIs and objects to native operating system objects ...")

a superstructure-dedicated operating system in communication with a device-native operating system and instantiating the application in the device in accordance with the superstructure and (col. 5:45-50 "... UI objects to native operating system ..." and col.5:55-65 "... transform the user interface APIs and objects to native operating system objects ...")

updating, information in a segment of the superstructure associated with the at least one application events responsive to receiving the at least one application event and independent of an update to a second segment in the superstructure (col. 12:40-50 "... convert source code statement ... capable of executing multiple operating system ..." and e.g. FIG. 3b, step 118 and related text)

Lewallen does not explicitly disclose updating the information in a segment of the superstructure associated. However, Gergic discloses a new application programming language which is based on user interaction with any device which a user is employing to access any type of information. Gergic teaches When the transcending is performed by a multi-modal/conversational browser (as described below), the gestures are uniquely identified using a node_id tag. This allows not only to produce the rendering in each registered modality (local or distributed), but also to provide very tight synchronization (i.e., on a gesture level or even sub-

gestures levels, when it is a gesture for which this makes sense). For example, an event (I/O event) immediately impacts the state of the dialogs (i.e., the state as maintained in the multi-modal shell,. In addition to the above Gericc teaches that upon successfully executing an update, the shell passes the newly updated application state to all registered browsers (paragraph [1312] and update the application state as each item of information is furnished (paragraph [1317]). Therefore it would have been obvious to one skilled in the art at the time the invention was made to update the application state to extract relevant information that can be present to the user during user interaction (paragraph [1321]).

Per claim 39, Lewallen discloses the method of claim 36 further comprising:
using the device-portable form as an intermediate or permanent storage format for recording data within the superstructure (col. 12:40-50 "... convert source code statement ... capable of executing multiple operating system ..." and e.g. FIG. 3b, step 118 and related text).

Per claim 40 (Currently amended), Lewallen discloses the method of any of claims 1 or 21 wherein the superstructure is organized into objects and classes (col. 8:60-67 "... data in any object accessible to the user interface including DOM tree..." and e.g. FIG. 3a, step 100 and related text)

Per claim 41 (Currently amended), Lewallen discloses the system of claim 38 wherein the platform-independent data superstructure further comprises at least one data structure that may

be interpolated when the device-native operating system requests data from the platform-independent data superstructure, (e.g. FIG. 1, 18 and related text).

Per claim 42 (Currently amended), Lewallen discloses the method of claim 3 wherein a first device can transmit to a second device a message containing an application event item, causing thereby cause the second device to place the application event item into a processing queue of the second device (e.g. FIG. 1, 18 and related text).

Per claim 43, Lewallen discloses the method of claim 20 wherein application logic can be distributed across the network by obtaining a portion of the logic from the remote device and transmitting it in a hierarchical form to the server without the necessity of adapting code therefor (e.g. FIG. 1, 18 and related text).

Per claim 44, Lewallen discloses the method of claim 20 further comprising providing updates to an application's state from the server to a remote device, by defining a minimal change set to the application's state and transferring it across the network from the server to the remote device, without the necessity of adapting code therefor (col. 12:40-50 "... convert source code statement ... capable of executing multiple operating system ..." and e.g. FIG. 3b, step 118 and related text).

Per claim 45 (Currently amended), Lewallen discloses the method of claim 1 further comprising incorporating media assets into the superstructure, for reference by running applications (e.g. FIG. 1 and related text).

Per claim 46 (Currently amended), Lewallen discloses the method of claim 1 further comprising incorporating by reference media assets outside the superstructure, for reference by running applications (e.g. FIG. 1 and related text).

Per claim 47 (Currently amended), Lewallen discloses the method of claim 1, wherein the step of receiving further comprises receiving by a wireless messaging device operable to communicate with a network serviced by a communications carrier, the platform-independent data superstructure enabling the creation, modification, and management of platform-independent user interfaces and associated display elements for an application having an appearance and behavior propagated with consistency across a network of heterogeneous platforms and communications carrier protocols (col. 12:40-50 “... convert source code statement ... capable of executing multiple operating system ...” and e.g. FIG. 3b, step 118 and related text)

the platform-independent data superstructure defining a user interface, maintaining a display state of the user interface, and storing an application state, program code and internal logic of the application (col. 5:45-50 “... UI objects to native operating system ...” and col. 5:55-65 “... transform the user interface APIs and objects to native operating system objects ...”)

Per claim 48 (Currently amended), Lewallen discloses the method of claim 47 further comprising the step of:

updating, in accordance with a superstructure segment update the application state and the user interface state on the wireless message device (col. 12:40-50 “... convert source code statement ... capable of executing multiple operating system ...” and e.g. FIG. 3b, step 118 and related text)

Per claim 50 (Currently amended), Lewallen discloses the method of claims 47 further comprising the step of updating in response to generated application events, a first segment of the superstructure associated with the application events independent of an update to a second segment in the superstructure, the application events including associated user interface events (col. 12:40-50 “... convert source code statement ... capable of executing multiple operating system ...” and e.g. FIG. 3b, step 118 and related text).

Per claim 52 (Currently amended), Lewallen discloses the method of claims 47 wherein the application includes a user interface, and wherein the user interface has a substantially identical appearance and behavior across heterogeneous devices, platforms or device-native operating system environments (col. 1:45-60 “... platform independent appearance and standard behavior...”).

Per claim 70 (Currently amended), Lewallen discloses the method of claim 47 further comprising the step of requesting, by at least one application events , a modification to the user interface (col. 12:40-50 “... convert source code statement ... capable of executing multiple

operating system ..." and e.g. FIG. 3b, step 118 and related text).

Per claim 71 (Currently amended), Lewallen discloses the method of claim 47 further comprising the step of requesting, by at least one application event, access to at least one template element stored in a library of platform-independent user interface templates provided by the platform-independent data superstructure (col.3:20-30 "... platform independent ... to allow Java application access operating system ...").

Per claim 72 (Currently amended), Lewallen discloses the method of claim 71 further comprising the step of requesting, by at least one application events at least one of an addition, subtraction replacement or other modification, to the at least one template element stored in a library of platform-independent user interface templates (col.3:20-30 "... platform independent ... to allow Java application access operating system ...").

Per claim 73 (Currently amended), Lewallen discloses the method of claim 74 further comprising the step of requesting, by at least one application event, an addition of user-defined content into the user interface (col.3:20-30 "... platform independent ... to allow Java application access operating system ...").

Per claim 75 (Currently amended), Lewallen discloses the method of claim 71 further comprising the step of enabling the creation of templates at a remote processor for subsequent representation in the superstructure and instantiation in the wireless device (e.g. FIG. 3a, step

136 and related text).

Per claim 76, Lewallen discloses the method of claim 75 wherein the remote processor is a personal computer (e.g. FIG. 1 and related text).

Per claim 79, Lewallen discloses the method of claim 47 further comprising configuring the user interface to enable a user to view, generate, send and manage messages (paragraph [0149] “... user interface related capabilities ...”).

Per claim 80, Lewallen discloses the method of claim 79 further comprising configuring the user interface to enable a user to generate messages containing any of text, images, sound, or other media content (e.g. FIG. 1, 10 and related text).

Per claim 81 (Currently amended), Lewallen discloses the method of claim 1 further comprising the steps of

executing, by the device, the application in accordance with the received superstructure (col.9:55-70 “... executing mixed statement ...”);

receiving, by the device, via a wireless communications channel accessible by a superstructure-based application environment, an application update, the application update including a data object operable to update a first segment of a platform-independent data superstructure in the superstructure-based application environment, independent of an update to a second segment in the platform-independent data superstructure col. 5:45-50 “... UI objects to

native operating system ..." and col.5:55-65 "... transform the user interface APIs and objects to native operating system objects ..."); and

receiving by the device a command to update, the application in accordance with the application update (col.6:15-25 "...mappings to transform W3C API interface in the mixed statement program ..."),

Per claim 82 (Currently amended), Lewallen discloses the method of claim 81 further comprising receiving by a plurality of devices broadcasted application updates and command to update an application (col. 12:40-50 "... convert source code statement ... capable of executing multiple operating system ..." and e.g. FIG. 3b, step 118 and related text)

Per claim 83 (Currently amended), Lewallen discloses the method of claim 1 further comprising the step of:

transmitting by the device to a plurality of a network via a wireless communication channel to at least one update (col.3:20-30 "... platform independent ... to allow Java application access operating system ...")

transmitting by the device to the plurlatly of devices in the network a command to update in the plurality of devices an executing application in accordance with received update (e.g. FIG. 3a, step 136 and related text)

Per claim 84 (Currently amended), Lewallen discloses the method of claim 83 wherein the step of transmitting the at least one update further comprises: transmitting by the device to a

plurality of devices in the network, via wireless communications channel, at least one update to a state of an executing application (e.g. FIG. 1 and related text)

Per claim 85, Lewallen discloses the method of any of claims 82, 83 or 84 further comprising:

ensuring that each device is in a consistent, known state at the time of broadcasting and that the update remains whole and complete (e.g. FIG. 3a, step 136 and related text)

Per claim 86, Lewallen discloses the method of any of claims 82, 83 or 84 further comprising: broadcasting, in an all-or-nothing manner, only complete segments of application update (col. 5:45-50 "... UI objects to native operating system ..." and col.5:55-65 "... transform the user interface APIs and objects to native operating system objects ...").

Per claim 87 (Currently amended), Lewallen discloses the method of claim 1 further comprising the steps of:

using an internal representation of the superstructure to store private data relating to requests from the application or the state or data type of a superstructure node (col. 8:60-67 "... data in any object accessible to the user interface including DOM tree..." and e.g. FIG. 3a, step 100 and related text), wherein the private data is not serialized when the application is paused, halted or migrated, and is stored in a manner conveniently accessible at application runtime, such that this non-conversational data is coherently recoverable so long as the private data can be re-

established upon de-serialization, based on public data that has been maintained in the superstructure (e.g. FIG. 1 and 202 and related text).

Per claim 89 (Currently amended), Lewallen discloses the method of claim 1 further comprising the steps of:

instantiating the application in the device (col. 5:45-50 “... UI objects to native operating system ...” and col.5:55-65 “... transform the user interface APIs and objects to native operating system objects ...”)

Per claim 91, this is the system version of the claimed method discussed above (Claim 1), wherein all claim limitations have been addressed and/or covered in cited areas as set forth above. Thus, accordingly, these claims are also obvious.

Per claim 92, this is the system version of the claimed method discussed above (Claim 2), wherein all claim limitations have been addressed and/or covered in cited areas as set forth above. Thus, accordingly, these claims are also obvious.

Per claim 93 (Currently amended), Lewallen discloses the system of claim 91 further comprising the steps of:

instantiating the application in the device (col. 5:45-50 “... UI objects to native operating system ...” and col.5:55-65 “... transform the user interface APIs and objects to native operating system objects ...”)

Per claim 94 (Currently amended), Lewallen discloses the system of claim 91 wherein the means for receiving further comprises means for receiving by the device, a platform-independent data superstructure defining the appearance and behavior of an application (col. 8:60-67 "... data in any object accessible to the user interface including DOM tree..." and e.g. FIG. 3a, step 100 and related text), the platform-independent data superstructure storing an application state, program code and internal logic of the application, and comprising an XML information superstructure (col. 5:1-10 "... arrangement of nodes ... XML document ...").

Per claim 95 (New), Lewallen discloses the method of claim 1, wherein the step of receiving the superstructure further comprises receiving, by the device, a platform-independent XML information superstructure defining the appearance and behavior of an application, the superstructure storing state, program code and internal logic of the application (col. 5:1-10 "... arrangement of nodes ... XML document ...").

Per claim 96 (New), Lewallen discloses the method of claim 1, wherein the step of receiving the superstructure further comprises receiving, by the device, a platform-independent data superstructure defining the appearance and behavior of an application, the superstructure serializable in whole or in part at any time and storing state, program code and internal logic of the application (col. 1:45-60 "... platform independent appearance and standard behavior...").

Per claim 97 (New), Lewallen discloses the method of claim 1 further comprising the step of encapsulating, by the platform-independent data superstructure, program code defining appearance and behavior of the application (col. 1:45-60 "... platform independent appearance and standard behavior...").

Per claim 98 (New), Lewallen discloses the method of claim 1, wherein the step of receiving the superstructure further comprises receiving, by the device, a platform-independent data superstructure comprising at least one data structure that may be interpolated when the device-native operating system requests data from the platform-independent data superstructure (col.3:20-30 "... platform independent ... to allow Java application access operating system ...").

Per claim 99 (New), Lewallen discloses the method of claim 21, wherein the step of receiving the superstructure further comprises receiving, by the device, a platform-independent XML information superstructure defining the appearance and behavior of an application, the superstructure storing state, program code and internal logic of the application (col. 1:45-60 "... platform independent appearance and standard behavior...").

Per claim 100 (New), Lewallen discloses the method of claim 21, wherein the step of receiving the superstructure further comprises receiving, by the device, a platform-independent hierarchical information superstructure defining the appearance and behavior of an application, the superstructure storing state, program code and internal logic of the application (col. 1:45-60 "... platform independent appearance and standard behavior...").

Per claim 101 (New), Lewallen discloses the method of claim 21 further comprising the step of encapsulating, by the platform-independent data superstructure, program code defining

appearance and behavior of the application (col. 1:45-60 "... platform independent appearance and standard behavior...").

Per claim 102 (New), Lewallen discloses the method of claim 21, wherein the step of receiving the superstructure further comprises receiving, by the device, a platform-independent data superstructure defining the appearance and behavior of an application, the superstructure serializable in whole or in part at any time and storing state, program code and internal logic of the application (e.g. FIG. 3a, step 108, 130 and 134 "method in native object class to determine pointer to node info and pointer to UI object in node info" and related text).

Per claim 103 (New), Lewallen discloses the method of claim 21, wherein the step of receiving the superstructure further comprises receiving, by the device, a platform-independent data superstructure comprising at least one data structure that may be interpolated when the device-native operating system requests data from the platform-independent data superstructure.

Per claim 104 (New), Lewallen discloses the system of claim 38, wherein the platform-independent data superstructure further comprises a platform-independent XML information superstructure (col. 5:1-10 "... arrangement of nodes ... XML document ...").

Per claim 105 (New), Lewallen discloses the system of claim 38, wherein the platform-independent data superstructure further comprises a platform-independent hierarchical information superstructure (e.g. FIG. 3a, step 108, 130 and 134 "method in native object class to determine pointer to node info and pointer to UI object in node info" and related text).

Per claim 106 (New), Lewallen discloses the system of claim 38, wherein the platform-independent data superstructure encapsulates program code defining appearance and behavior of the application (col. 1:45-60 "... platform independent appearance and standard behavior...").

Per claim 107 (New), Lewallen discloses the system of claim 38, wherein the platform-independent data superstructure is serializable in whole or in part at any time (col. 1:45-60 "... platform independent appearance and standard behavior...").

Per claim 108 (New), Lewallen discloses the system of claim 91, wherein the means for receiving further comprises means for receiving, by a device, a platform-independent XML information superstructure (col. 5:1-10 "... arrangement of nodes ... XML document ...").

14. Claims 77 and 78 are rejected under 35 U.S.C. 103(a) as being unpatentable over Lewallen (US 6,854,123 B1) in view of Geric et al. (US 2003/0046316 A1) further in view of Snyder (US 6,707,475 B1).

Neither Lewallen nor Geric explicitly disclose configuring the user interface to respond to controls adapted to be actuated by a user's thumbs and configuring the user interface to provide visual, sonic, tactile or other human-perceptible indications in response to commands entered by a user, or other application events.. However, Snyder teaches user interface includes a cursor control device having a touch-pad device with thumb actuation switch located on its side. When employing the device, the user resets a hand on a built-in palm rest to stabilize the hand, positions the fingertip for pointing, and positions the thumb for clicking (col. 5: 25-35). Therefore it would have been obvious to one skilled in the art at the time of the invention was

made to combine Lewallen, Geric and Snyder to select data captured by the cursor for selecting and displaying navigational information as once suggested by Snyder (col.2: 15-30).

Response to Arguments

15. Applicant's arguments with respect to claims 1-48, 50, 52, 70-73, 75-87, 89 and 91-108 have been considered but are moot in view of the new ground(s) of rejection. See Lewallen (US 6,854,123) and Geric et al. (US 2003/0046316 A1)

Conclusion

16. Applicant's amendment necessitated the new ground(s) of rejection presented in this Office action. Accordingly, **THIS ACTION IS MADE FINAL**. See MPEP § 706.07(a). Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).

A shortened statutory period for reply to this final action is set to expire THREE MONTHS from the mailing date of this action. In the event a first reply is filed within TWO MONTHS of the mailing date of this final action and the advisory action is not mailed until after the end of the THREE-MONTH shortened statutory period, then the shortened statutory period will expire on the date the advisory action is mailed, and any extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no event, however, will the statutory period for reply expire later than SIX MONTHS from the date of this final action.

Any inquiry concerning this communication or earlier communications from the examiner should be directed to ISAAC T. TECKLU whose telephone number is (571)272-7957. The examiner can normally be reached on M-TH 9:300A - 8:00P.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Tuan Q. Dam can be reached on (571) 272-3695. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see <http://pair-direct.uspto.gov>. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a USPTO Customer Service Representative or access to the automated information system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

/Isaac T Tecklu/

Examiner, Art Unit 2192

/Tuan Q. Dam/

Supervisory Patent Examiner, Art Unit 2192